

THE LOST MELODY PHENOMENON

MERLIN CARL

ABSTRACT. A typical phenomenon for machine models of transfinite computations is the existence of so-called lost melodies, i.e. real numbers x such that the characteristic function of the set $\{x\}$ is computable while x itself is not (a real having the first property is called recognizable). This was first observed by J. D. Hamkins and A. Lewis for infinite time Turing machine (see [HaLe]), then demonstrated by P. Koepke and the author for *ITRMs* (see [ITRM]). We prove that, for unresetting infinite time register machines introduced by P. Koepke in [wITRM], recognizability equals computability, i.e. the lost melody phenomenon does not occur. Then, we give an overview on our results on the behaviour of recognizable reals for *ITRMs* as introduced in [KoMi]. We show that there are no lost melodies for ordinal Turing machines (*OTMs*) or ordinal register machines (*ORMs*) without parameters and that this is, under the assumption that $0^\#$ exists, independent from *ZFC*. Then, we introduce the notions of resetting and unresetting α -register machines and give some information on the question for which of these machines there are lost melodies.

1. INTRODUCTION

The research on machine models of transfinite computations began with the seminal Hamkins-Lewis paper [HaLe] on Infinite Time Turing Machines (*ITTM*s). These machines, which are basically classical Turing machines equipped with transfinite running time, have successfully been applied to various areas of mathematics such as descriptive set theory ([Co], [SeSc]) and model theory ([HMSW]) and turned out to show a variety of fascinating behaviour. A particularly interesting feature that has frequently played a role in applications is the existence of so-called lost melodies. A lost melody is a real number $x \subseteq \omega$ which is recognizable, i.e. for some *ITTM*-program P , the computation of P with y on the input tape (which plays the role of a real oracle for an *ITTM*) is defined for all y and outputs 1 iff $y = x$ and otherwise outputs 0, but not computable, i.e. no program computes the characteristic function of x . The existence of lost melodies for *ITTM*s was observed and proved in [HaLe].

In the meantime, a rich variety of transfinite machine types have been defined, studied and related to each other: Koepke introduced Infinite Time Register Machines (see [wITRM]), which were later relabeled as unresetting or weak Infinite Time Register Machines (*wITRM*s) when

an enhanced version was considered in [ITRM]. Further generalizations led to α -Turing machines ([KoSe1]) α - β -Turing machines, transfinite λ -calculus ([Se]), the hypermachines of Friedman and Welch (basically *ITTM*s with a more complex limit behaviour, see [FrWe]) and infinite time Blum-Shub-Smale-machines ([KoSe2]). An arguably ultimate upper bound is set by Koepke's ordinal register machines (*ORM*s) and ordinal Turing machines (*OTM*s), which, using ordinal parameters, can calculate the whole of Gödel's constructible hierarchy L . ([ICTT] contains an argument to the effect that *OTM*-computability is indeed a conceptual analogue of Turing-computability in the transfinite.) For many of these machine types, the computational strength has been precisely determined.

In this paper, we are interested in the question how typical the existence of lost melodies is for models of transfinite computations. While it was shown in [ITRM] that *ITRM*s, like *ITTM*s, have lost melodies, the question was to the best of our knowledge not considered for any other of these machine types and has in particular been open concerning *wITRM*s. Specifically, we focus on machine models generalizing register machines: In section 1, we prove that there are no lost melodies for unresetting *ITRM*s, we summarize (mostly leaving out or merely sketching proofs) in section 2 some of our earlier results on *ITRM*-recognizability obtained in [Ca] and [Ca2] and proceed in section 3 to show that there are again no lost melodies for ordinal register- and Turing machines, without ordinal parameters and that the answer for ordinal machines with parameters is undecidable under a certain set-theoretical assumption. Then, for the parameter-free case, we interpolate between these extrem cases by introducing resetting and unresetting α -register machines and show that for resetting α -register machines, lost melodies always exist. For unresetting α -register machines, the picture is quite different: It turns out that, while there are no lost melodies for $\alpha = \omega$, there exist countable values of α for which there are, but their supremum is countable, so that from some $\gamma < \omega_1$ on, lost melodies for unresetting α -register machines cease to exist.

Let us now introduce the relevant machine types, the resetting and unresetting α -register machines. (The unresetting version was originally suggested in the final paragraph of [wITRM].) An α -register machine has finitely many registers, each of which can store a single ordinal $< \alpha$. The instructions for an α -register machine (also simply called α -machine) are the same as for the unlimited register machines of [Cu]: the increasing of a register content by 1, copying a register content to another register, reading out the r_i th bit of an oracle (where r_i is the content of the i th register), jumping to a certain program line provided a certain register content is 0, and stopping. Programs for α -register machines are finite sequences of instructions, as usual. The running time of an α -machine is the class of ordinals. At successor

times, computations proceed as for the classical model of unlimited register machines, introduced in [Cu]. It remains to fix what to do at a limit time λ . We consider three possibilities, where Z_ι denotes the active program line at time ι and $R_{i\iota}$ denotes the content of the i th register at time ι :

- Z_λ and $R_{i\lambda}$ are undefined. Setting $\lambda = \omega$, this would just be a classical *URM*
- $Z_\lambda := \liminf_{\iota < \lambda} Z_\iota$, $R_{i\lambda} = \liminf_{\iota < \lambda} R_{i\iota}$, if the latter is $< \alpha$ and otherwise, the computation is undefined. Setting $\alpha = \omega$, these are the unresetting or weak infinite time register machines introduced in [wITRM].¹ We call these unresetting or weak α -machines.
- $Z_\lambda := \liminf_{\iota < \lambda} Z_\iota$, $R_{i\lambda} = \liminf_{\iota < \lambda} R_{i\iota}$, if the latter is $< \alpha$ and otherwise $R_{i\lambda} = 0$. Setting $\alpha = \omega$, these are the infinite time register machines (*ITRMs*) of [KoMi]. We call these resetting or strong α -machines.

Most of our notation and terminology is standard. *KP* is Kripke-Platek set theory (see e.g. [Sa]), *ZF*[−] is Zermelo-Fraenkel set theory without the power set axioms in the version described in [GHJ]. If P is a program and x a real, then $P^x \downarrow$ means that P , when run in the oracle x , stops, while $P^x \uparrow$ means that P in the oracle x diverges. $P^x \downarrow = y$ means that $P^x(i) \downarrow$ for all $i \in \omega$ and that in the final state of $P^x(j)$, the first register contains 1 iff $j \in y$ and otherwise 0. We write $x \leq_h y$ for hyperarithmetic reducibility, i.e. for $x \in L_{\omega_1^{CK,y}}[y]$. On denotes the class of ordinals, small greek letters denote ordinals unless stated otherwise. $p : On \times On \rightarrow On$ is Cantor's pairing function.

It turns out (see [KoMi]) that unresetting ω -machines are much weaker than their resetting analogue; in particular, resetting ω -machines can compute all finite iterations of the halting problem for unresetting ω -machines. We fix the following general definitions:

Definition 1. Let P be program of any of the machine types described above, and let x be a real. We say that P recognizes x iff $P^x \downarrow = 1$ and $P^y \downarrow = 0$ for all $y \neq x$. We say that x is recognizable by an (un)resetting α -machine iff there is a program P for such a machine that recognizes x . When the machine type is clear from the context, we merely state that x is recognizable.

2. WEAK ITRMS

Proposition 2. Let x be *wITRM*-computable. Then x is *wITRM*-recognizable.

¹In the cited paper, these machines are just called infinite time register machines, without further qualification. Later on, when resetting infinite time register machines were introduced, the terminology was changed.

Proof. Let P be a $wITRM$ -program that computes x . The idea is to compare x to the oracle bitwise. A bit of care is necessary to arrange this comparison without overflowing registers. Use a separate counting register R and two flag registers R_1^{flag} and R_2^{flag} . Initially, R and R_1^{flag} contain 0 and R_2^{flag} contains 1. In a computation step, when R contains i , compute the i th bit of x and compare it to the oracle. If these bits disagree, we stop with output 0. Otherwise, we successively set all registers but R_1^{flag} and R_2^{flag} to 0 once and then set the content of R to $i + 1$ (after a register has been set to 0, it may be used to store i for this purpose) and swap the contents of R_1^{flag} and R_2^{flag} . In this way, if the number in the oracle is x , then a state will eventually occur in which R_1^{flag} and R_2^{flag} both contain 0 and R contains 0, in which case we output 1. \square

Definition 3. Let us denote by $wRECOG$ the set of reals recognizable by a weak $ITRM$ and by $wCOMP$ the set of reals computable by a weak $ITRM$.

The following is the relativized version of Theorem 1 of [wITRM]:

Theorem 4. Let $x, y \subseteq \omega$. Then x is $wITRM$ -computable in the oracle y iff $x \in L_{\omega_1^{CK,y}}[y]$. In particular, x is $wITRM$ -computable iff $x \in L_{\omega_1^{CK}}$ iff x is hyperarithmetic.

Proof. The proof given in [ITRM] relativizes. We omit the proof to avoid what would amount to a mere repetition of that proof. \square

Lemma 5. Let $x \subseteq \omega$ and let $M \models KP$ be such that $\omega^M = \omega$ and $x \in M$. Then $\omega_1^{CK,x}$ is an initial segment of On^M .

Proof. If $x \in M$, then, as $M \models KP$, we have $z \in M$ for every z which is recursive in x .

Now let $x \in M$. We have to show that every $\alpha < \omega_1^{CK,x}$ belongs to the well-founded part of M . Since $M \models KP$, M satisfies the recursion theorem for Σ_1 -definitions. Let $z \subset \omega \times \omega$ be such that (ω, z) is a well-ordering. For all $\beta \in On$, we define, by Σ_1 -recursion, a function F via $F(\beta) = \sup_z \{F(\gamma) + 1 \mid \gamma < \beta\}$ if this supremum exists, and otherwise $F(\beta) = \omega$.

We show that $\text{rng}(F) = \omega$: Otherwise, we have $\text{rng}(F) \subsetneq \omega$ and $\text{rng}(F)$ is closed under z -predecessors. Since (ω, z) is a well-ordering in V , $\text{rng}(F)$ must have a z -supremum $n \in \omega$. Hence $\text{rng}(F) = \{m \in \omega \mid (m, n) \in z\} \in M$: By the injectivity of F , F^{-1} is Σ_1 -definable. By Σ_1 -replacement, $\text{rng}(F^{-1})$ is a set, hence an ordinal γ . Consequently, we have $\omega \subseteq \text{rng}(F)$. We now show that $\text{rng}(F|_\gamma) = \omega$ for some $\gamma \in On \cap M$. Suppose that $\omega \notin \text{rng}(F)$. Then F is injective and $F^{-1} : \omega \rightarrow On$ is a function, contradicting Σ_1 -replacement (as $On \cap M$ is not a set in M). Relativizing this argument to x , we obtain the desired result. \square

Lemma 6. Let P be a $wITRM$ -program, and let $x \subseteq \omega$. Then $P^x \uparrow$ iff there exist $\sigma < \tau < \omega_1^{CK,x}$ such that $Z(\tau) = Z(\sigma)$, $R_i(\tau) = R_i(\sigma)$ for all $i \in \omega$ and $R_i(\gamma) \geq R_i(\sigma)$ for all $i \in \omega$, $\sigma < \gamma < \tau$. (Here, $Z(\gamma)$ and $R_i(\gamma)$ denote the active program line and the content of register i at time γ .)

Proof. This is an easy adaption of Lemma 3 of [KoMi]. \square

The following lemma allows us to quantify over countable ω -models of KP by quantifying over reals:

Lemma 7. There is a Σ_1^1 -statement $\phi(v)$ such that $\phi(x)$ holds only if x codes an ω -model of KP and such that, for any countable ω -model M of KP , there is a code c for M such that $\phi(c)$ holds.

Proof. Every countable ω -model M of KP can be coded by a real $c(M)$ in such a way that the $i \in \omega$ is represented by $2i$ in c and ω is represented by 1. We can then consider a set S of statements saying that a real c codes a model of KP together with $\{P_k \mid k \in \omega + 1\}$, where P_k is the statement $\forall i < k(p(2i, 2k) \in c) \wedge \forall j \exists i < k(p(j, 2k) \in c \rightarrow j = 2i)$ for $k \in \omega$ and P_ω is the statement $\forall i(p(2i, 1) \in c) \wedge \forall j \exists i(p(j, 1) \in c \rightarrow j = 2i)$. Then $\bigwedge S$ is a hyperarithmetic conjunction of arithmetic formulas in the predicate c . But such a conjunction is equivalent to a Σ_1^1 -formula. \square

Theorem 8. Let x be recognizable by a $wITRM$. Then $\{x\}$ is a Σ_1^1 -singleton.

Proof. Let P be a program that recognizes x on a $wITRM$. Let $KP(z)$ be a Σ_1^1 -formula (in the predicate z) stating that z codes an ω -model of KP with ω represented by 1 and every integer i represented by $2i$ as constructed in Lemma 7. Let $E(y, z)$ be a first-order formula (in the predicates y and z) stating that the structure coded by z contains y . (We can e.g. take $E(y, z)$ to be $\exists k \forall i(z(i) \leftrightarrow z(p(2i, k)))$.) Furthermore, let $Acc_P(z, y)$ be a first-order formula (in the predicates y and z) stating that $P^y \downarrow = 1$ in the structure coded by z . Finally, let $NC_P(y)$ be a first-order formula (in the predicate y) stating that in the computation P^y , there are no two states s_{ι_1}, s_{ι_2} with $\iota_1 < \iota_2$ such that $s_{\iota_1} = s_{\iota_2}$ and, for every $\iota_1 < \iota < \iota_2$, the content r_{ι_i} of register R_i at time ι is at least $r_{\iota_{i1}}$ (the content of R_i at time ι_1) and the index of the active program at time ι is not smaller than the index of the active program line at time ι_1 . (This is just the cycle criterion from Lemma 6.)

This is possible in KP models containing x since, by Lemma 5 above, $\omega_1^{CK,x}$ is an initial segment of the well-founded part of each such model and, by Lemma 6, the computation either cycles before $\omega_1^{CK,x}$ or stops - thus the cycling or halting behaviour takes part in the well-founded part of the model and is hence absolute between such a model and

V. Now, take $\phi(a)$ to be $\exists z(KP(z) \wedge E(a, z) \wedge Acc_P(z, a) \wedge NC_P(a))$. This is a Σ_1^1 -formula. We claim that x is the only solution to $\phi(a)$: To see this, first note that x clearly is a solution, since $\omega_1^{CK, x}$ is an initial segment of every KP -model containing x by Lemma 5.

On the other hand, assume that $b \neq x$. In this case, as P recognizes x , we have $P^b \downarrow = 0$ in the real world, and hence, by absoluteness of $wITRM$ -(oracle)-computations for KP -models containing the relevant oracles, also inside $L_{\omega_1^{CK, b}}[b]$. Now $L_{\omega_1^{CK, b}}[b]$ is certainly a countable KP -model containing b , hence a counterexample to $\phi(b)$. Hence $\phi(b)$ is false if $b \neq x$, as desired. \square

Corollary 9. If a real x is $wITRM$ -recognizable, then it is $wITRM$ -computable. Hence, there are no lost melodies for weak $ITRM$ s.

Proof. By Kreisel's basis theorem (see [Sa], p. 75), if a is not hyperarithmetical and $B \neq \emptyset$ is Σ_1^1 , then B contains some element b such that $a \not\leq_h b$. Now suppose that x is $wITRM$ -recognizable. By Theorem 8, $\{x\}$ is Σ_1^1 and certainly non-empty. If x was not hyperarithmetical, then, by Kreisel's theorem, $\{x\}$ would contain some b such that $x \not\leq_h b$. But the only element of $\{x\}$ is x , so $x \notin HYP$ implies $x \not\leq_h x$, which is absurd. Hence $x \in HYP$. So x is $wITRM$ -computable. \square

3. RESETTING ITRMS

$ITRM$ -recognizability was considered in [ITRM], [Ca] and [Ca2]. We give here a summary of some of the most important results. Recall that an $ITRM$ is different from a $wITRM$ in that it, in case of a register overflow, resets the content of the overflowing registers to 0 and continues computing.

The following characterization of the computational strength of $ITRM$ s with real oracles is a relativized version of the main theorem of [KoMi]:

Theorem 10. x is $ITRM$ -computable in the oracle y iff $x \in L_{\omega_\omega^{CK, y}}[y]$.

We saw that computability equals recognizability for $wITRM$ s. For $ITRM$ s, the situation is very different. Clearly, analogous to Proposition 2, the computable reals are still recognizable. But, for $ITRM$ s, the lost melody phenomenon does occur:

Theorem 11. There exists a real x such that x is not $ITRM$ -computable, but $ITRM$ -recognizable.

Proof. x can be taken to be a $<_L$ -minimal real coding an \in -minimal L_α such that $L_\alpha \models ZF^-$. See [ITRM] for the details. \square

Remark 12. There are more straightforward examples. In [Ca2], it is shown that, if $(P_i | i \in \omega)$ is some natural enumeration of the $ITRM$ -programs, then $h := \{i \in \omega | P_i \downarrow\}$, the halting number for $ITRM$ s, is recognizable. The usual argument of course shows that it is not $ITRM$ -computable.

Definition 13. $RECOG$ denotes the set of $ITRM$ -recognizable reals.

Remark 14. Write $RECOG_n$ for the set of reals that are recognizable by an $ITRM$ using at most n registers. It was shown in [Ca] that $RECOG_n \subsetneq RECOG$, i.e. the recognizability strength of $ITRM$ s increases with the number of registers. This corresponds to the result established in [KoMi] that the computational strength of $ITRM$ s increases with the number of registers.

Using Shoenfield's absoluteness lemma, it is not hard to see that recognizable reals are always constructible (see [Ca]). We consider the distribution of recognizable reals in the canonical well-ordering $<_L$ of the constructible universe:

Theorem 15. There are gaps in the $ITRM$ -recognizable reals, i.e. there are $x, y, z \in \mathcal{P}(\omega) \cap L$ such that $x <_L y <_L z$, $x, z \in RECOG$, but $y \notin RECOG$.

Proof. As there are only countably many $ITRM$ -recognizable real, there must exist a countable α such that $L_\alpha \models ZF^-$ and L_α contains some non-recognizable reals y . Let z be the $<_L$ -minimal code of the \in -minimal L_α with these properties and let $x = 0$. Then x, y and z are as desired. The details can be found in [Ca]. \square

This suggest the detailed study of the distribution of the $ITRM$ -recognizable reals among the constructible reals, which was carried out in [Ca] and [Ca2]. We summarize the main results.

Definition 16. Let $\alpha \in On$. α is called Σ_1 -fixed iff there exists a Σ_1 -formula ϕ such that α is minimal with $L_\alpha \models \phi$. We also let $\sigma := \sup\{\alpha \mid \alpha \text{ is } \Sigma_1\text{-fixed}\}$.

Remark 17. It is easy to see by reflection that the Σ_1 -fixed ordinals are countable and that there are countably many of them (as there are only countable many formulas). Hence σ is countable as well. It can also be shown that σ is the supremum of parameter-free OTM -halting times.

Theorem 18. (a) $RECOG \subseteq L_\sigma$
 (b) $\{\alpha \mid RECOG \cap (L_{\alpha+1} - L_\alpha) \neq \emptyset\}$ is cofinal in σ
 (c) For every $\gamma < \sigma$, there exists $\alpha < \sigma$ such that $RECOG \cap (L_{\alpha+\gamma} - L_\alpha) = \emptyset$.

Proof. See [Ca]. \square

The $ITRM$ -computability of a real can be effectively characterized in purely set theoretical terms (namely as being an element of $L_{\omega_\omega^{CK}}$). Correspondingly, we have the following necessary criterion for $ITRM$ -recognizability:

Lemma 19. Let $x \in \text{RECOG}$. Then $x \in L_{\omega_\omega^{CK,x}}$. In particular, we have $\omega_\omega^{CK,x} > \omega_\omega^{CK}$, hence $\omega_i^{CK,x} > \omega_i^{CK}$ for some $i \in \omega$.

Proof. See [Ca2]. \square

Lemma 19 in fact allows a machine-independent characterization of recognizability:

Theorem 20. Let $x \in \mathcal{P}^L(\omega)$. Then $x \in \text{RECOG}$ iff x is the unique witness for some Σ_1 -formula in $L_{\omega_\omega^{CK,x}}$.

Proof. See [Ca2]. \square

One might now ask where non-recognizable occur; clearly, every real in $L_{\omega_\omega^{CK}}$ is recognizable, but what happens above ω_ω^{CK} ? E.g., is there some $\alpha > \omega_\omega^{CK}$ such that the reals in L_α are still all recognizable? It turns out that this is not the case and that, in fact, unrecognizables turn up wherever possible in the L -hierarchy.

Definition 21. $\alpha \in \text{On}$ is an index iff $(L_{\alpha+1} - L_\alpha) \cap \mathcal{P}(\omega) \neq \emptyset$.

Theorem 22. Let $\alpha \geq \omega_\omega^{CK}$ be an index. Then there exists a real $x \notin \text{RECOG}$ such that $x \in L_{\alpha+1} - L_\alpha$.

Proof. See [Ca2]. \square

In the light of Lemma 19, it is natural to concentrate the study of recognizability on reals x with $x \in L_{\omega_\omega^{CK,x}}$. It turns out that the distribution of recognizables becomes much tamer when we do this:

Theorem 23. (The ‘All-or-nothing-theorem’) Let γ be an index. Then either all $x \in L_{\gamma+1} - L_\gamma$ with $x \in L_{\omega_\omega^{CK,x}}$ are recognizable or none of them is.

Proof. See [Ca2]. The idea is that, given a recognizable $a \in L_{\gamma+1} - L_\gamma$, this can be used to identify the $<_L$ -minimal code c of $L_{\gamma+1}$, which can in turn be used to identify every real in $L_{\gamma+1}$. \square

4. ORDINAL MACHINES

Ordinal Turing machines (*OTMs*) and ordinal register machines (*ORMs*) were introduced in [OTM] and [ORM], respectively, and seem to provide an upper bound on the strength of a reasonable transfinite model of computation. (See e.g. [ICTT] for an argument in favor of this claim.) In the papers just cited, Koepke proves that, when finite sets of ordinals are allowed as parameters, these machines can compute the characteristic function of a set x of ordinals iff $x \in L$. In particular a real x is computable by such a machine iff $x \in L$. We formulate our results from now on for *OTMs* only, as they carry over verbatim to *ORMs*. To clarify the role of the parameters, we give a separate definition for recognizability by *OTMs* with parameters.

Definition 24. $x \subseteq \omega$ is parameter-OTM-recognizable iff, for some OTM-program P with a finite sequence $\vec{\gamma}$ of ordinal parameters and every $y \subseteq \omega$, $P^y \downarrow = 1$ iff $x = y$ and otherwise $P^y \downarrow = 0$.

In the constructible universe, there are no lost melodies for parameter-OTMs:

Lemma 25. Assume that $V = L$ and let x be parameter-OTM-recognizable. Then x is parameter-OTM-computable.

Proof. By [Ko], a set S of ordinals is computable by an OTM-program with ordinal parameters iff $S \in L$. Hence, every constructible real is parameter-OTM-computable, and in particular each parameter-OTM-recognizable real. \square

Note that, of course, every constructible real is also parameter-OTM-recognizable.

Lemma 26. Assume that $\omega_1^L = \omega_1$. Let $\gamma < \omega_1^L$ and suppose that $x \subseteq \omega$ is recognizable by some program P in the parameter γ . Then $x \in L$.

Proof. As γ is countable in L , there is a constructible real z coding γ . Pick $z <_L$ -minimal. Then $\exists y P^y(\gamma) \downarrow = 1$ is expressible by a Σ_1 -formula in the parameter z . Let ρ be the running time of $P^x(\gamma)$ (i.e. the length of the computation). Then ρ is countable: To see this, let c be the computation of $P^x(\gamma)$, κ a cardinal in $L[x]$ such that $\kappa > \max\{|c|, \aleph_1^{L[x]}\}$ and consider in $L_\kappa[x]$ the Σ_1 -Skolem hull H of $\{c, \gamma, x\}$. By condensation in $L[x]$, there is some $\bar{\kappa}$ such that H collapses to $L_{\bar{\kappa}}[x]$; let $\pi : H \rightarrow L_{\bar{\kappa}}[x]$ be the collapsing map. Moreover, as H is countable, so is $L_{\bar{\kappa}}[x]$. As $x \subseteq \omega \subseteq H$, we have $\pi(x) = x$. As $\omega + 1 \subseteq H$, $\gamma \in H$ and H contains a bijection between ω and γ , we have $\gamma \subseteq H$, so $\pi(\gamma) = \gamma$. As ‘ c is the computation of $P^x(\gamma)$ ’ is expressible by a Σ_1 -formula, $L_{\bar{\kappa}}[x]$ believes that $\pi(c)$ is the computation of $P^{\pi(x)}(\pi(\gamma))$. As $L_{\bar{\kappa}}[x]$ is transitive and by absoluteness of computations, $\pi(c)$ really is the computation of $P^{\pi(x)}(\pi(\gamma))$. As $\pi(x) = x$ and $\pi(\gamma) = \gamma$, we have $\pi(c) = c$, so $c \in L_{\bar{\kappa}}[x]$; as the latter is transitive and countable, c is countable. Hence ρ is countable.

As $\rho < \omega_1 = \omega_1^L$, ρ is countable in L . As there are cofinally in ω_1^L many admissible ordinals, let $\alpha > \max\{\gamma, \rho\}$ be a limit of admissible ordinals which is also a limit of index ordinals such that $z \in L_\alpha$. Now $\exists y P^y(\gamma) \downarrow = 1$ is expressible as a Σ_1 -formula $\phi(z)$ in the real parameter z . As $\rho < \alpha$ and $x \in L_\alpha[x]$, $\phi(z)$ holds in $L_\alpha[x]$ and hence in V_α . By a theorem of Jensen and Karp (see section 5 of [JeKa]), Σ_1 -formulas are absolute between L_ζ and V_ζ when ζ is a limit of admissibles and L_ζ contains the relevant parameters. Hence $\phi(z)$ holds in L_α . So L_α contains a real y such that, in L_α , we have $P^y(\gamma) \downarrow = 1$. By absoluteness of computations, we have $P^y(\gamma) \downarrow = 1$ in the real world. As x

is recognized by P in the parameter γ , it follows that $y = x$. Hence $x \in L$. \square

On the other hand, if the universe is much unlike L and we allow uncountable parameters, lost melodies for parameter-OTMs can occur:

Theorem 27. Assume that 0^\sharp exists. Then there is a lost melody for parameter-OTMs. In fact, 0^\sharp is parameter-OTM-recognizable in the parameter ω_1 .

Proof. By Theorem 14.11 of [Ka], the relation $x = 0^\sharp$ is Π_2^1 , so $x \neq 0^\sharp$ is Σ_2^1 . Furthermore, Σ_2^1 -relations are absolute between transitive models of KP containing ω_1 (see e.g. Corollary 1 of [SeSc]). Now, let $\alpha > \omega_1$ be minimal such that $M := L_\alpha[0^\sharp] \models KP$. Then $L[0^\sharp]$ contains a bijection $f : \omega_1 \leftrightarrow M$. Hence, M is coded by $r := \{p(\iota_1, \iota_2) \mid \iota_1, \iota_2 < \omega_1 \wedge f(\iota_1) \in f(\iota_2)\} \in L[0^\sharp]$. To recognize 0^\sharp with an OTM when ω_1 is given as a parameter, we proceed as follows: Given a real x in the oracle, search through the subsets of ω_1 in $L[x]$ (by a similar procedure used in the proof of Theorem 29) for a set c coding a KP -model M' of the form $L_\beta[x]$ that contains ω_1 . As such sets exist in $L[x]$, such a c will eventually be found. Once this has happened, check, using c , whether $M' \models x = 0^\sharp$. If not, then, by absoluteness, $x \neq 0^\sharp$, otherwise $x = 0^\sharp$. \square

Taken together, the last two theorems readily yield:

Corollary 28. If 0^\sharp exists, then it is undecidable in ZFC whether there are lost melodies for parameter-OTMs.

From now on, when we talk about OTMs, we always mean the parameter-free case. What happens if we consider OTMs without ordinal parameters? It turns out that then, there are no lost melodies:

Theorem 29. Let $x \subseteq \omega$ and P be an OTM-program such that, for each $y \subseteq \omega$, we have $P^y \downarrow = 1$ iff $y = x$ and $P^y \downarrow = 0$, otherwise. Then x is OTM-computable (without parameters).

Proof. In [Ko], it is shown that every constructible set of ordinals is uniformly computable from an appropriate finite set of ordinal parameters. Hence, there is a program Q which, for every input $\vec{\alpha}$, a finite sequence of ordinals, computes the characteristic function of a set x of ordinals in a such a way that for every constructible $x \subseteq On$, there exists $\vec{\gamma}_x$ such that Q computes the characteristic function of x on input $\vec{\gamma}_x$. We will use Q to search through the constructible reals, looking for some $x \subseteq On$ such that $P^{x \cap \omega} \downarrow = 1$. To do this, we use some natural enumeration $(\vec{\gamma}_\iota \mid \iota \in On)$ of finite sequences of ordinals and carry out the following procedure for each $\iota \in On$. First, find $\vec{\gamma}_\iota$, and let x_ι be the set of ordinals whose characteristic function is computed by Q on input $\vec{\gamma}_\iota$. Then check, using P , whether $P^{x_\iota \cap \omega} \downarrow = 1$. As $P^y \downarrow$ for

all $y \subseteq \omega$, this will eventually be determined. If $P^{x_\iota \cap \omega} \downarrow = 1$, then x is found and we can write it on the tape. Otherwise, continue with $\iota + 1$. In this way, every constructible real will eventually be checked. By Shoenfield's absoluteness theorem, x must be constructible, hence x will at some point be considered, identified and written on the tape. Thus x is computable. \square

In fact, by almost the same reasoning, a much weaker assumption on x is sufficient:

Corollary 30. Let $x \subseteq \omega$ and P be an *OTM*-program such that, for each $y \subseteq \omega$, we have $P^y \downarrow$ iff $y = x$ and $P^y \uparrow$, otherwise. Then x is *OTM*-computable (without parameters).

Proof. First, observe that, by Shoenfield absoluteness, such an x must be an element of L . Now, we use a slight modification of the proof of Theorem 29: Again, we use a program Q to successively write all constructible sets of naturals to the tape. But now, we let P run simultaneously on all the written reals. At some point, x will be written to the tape and at some later point, P will halt on it. When that happens, just copy the real on which P halted to the beginning of the tape, thus writing x . This can then be used to decide every bit of x . \square

Remark 31. An easy reflection argument shows that a halting *OTM*- (and *ORM*-) computation with a real oracle always has a countable running time. Our results above hence in fact hold for unresetting ω_1 -machines as well.

In the parameter-free case, this shows that, for extremely strong models of computation, the lost melody phenomenon is no longer present. This motivates a further inspection what exactly is necessary for the existence of lost melodies.

5. α -REGISTER MACHINES

Recall that, for $\alpha \in On$, let a resetting/unresetting α -register machine works like an *ITRM*/*wITRM* with the difference that a register may now contain an arbitrary ordinal $< \alpha$. Hence, an *ITRM* is a resetting ω -register machine and a *wITRM* is an unresetting ω -register machine. This generalization was suggested at the end of [wITRM].

We denote by $wCOMP_\alpha$, $COMP_\alpha$, $wRECOG_\alpha$ and $RECOG_\alpha$ the set of reals computable by an unresetting α -register machine, computable by a resetting α -register machine, recognizable by an unresetting α -register machine and recognizable by a resetting α -register machine, respectively.

We have seen that $wCOMP_\omega = wRECOG_\omega$, $COMP_\omega \subsetneq RECOG_\omega$, and that lost melodies for unresetting machines vanish when the register contents are unbounded. Hence, we ask:

For which α are there lost melodies for resetting/unresetting α -register machines?

We start with the following easy observation:

Lemma 32. (1) Let $\alpha \geq \omega$. Then $wCOMP_\alpha \subseteq wRECOG_\alpha$ and $COMP_\alpha \subseteq RECOG_\alpha$.
 (2) For all α , we have $wCOMP_\alpha \subseteq COMP_\alpha$ and $wRECOG_\alpha \subseteq RECOG_\alpha$.

Proof. (1) As $\alpha \geq \omega$, we can again compute a real x and compare it to the oracle bitwise.

(2) A terminating computation by an unresetting α -machine will run exactly the same on a resetting α -machine. \square

Lemma 33. Let $\alpha > \beta$ be ordinals. Assume that there is an unresetting α -program P such that $P(b) \downarrow = 1$ iff $b = \beta$ and $P(b) \downarrow = 0$, otherwise. Then $COMP_\beta \subseteq wCOMP_\alpha$.

Proof. Given α, β and P as in the assumptions, let $y \in COMP_\beta$, and let Q be a resetting β -program computing y . To compute y on an unresetting α -machine, we describe how to simulate Q on such a machine. Assume that Q uses k registers. Reserve k registers R_1^Q, \dots, R_k^Q of the unresetting α -machine. Then, we proceed as follows: At successor steps, simply carry out Q on R_1^Q, \dots, R_k^Q . At limit steps of the Q -computation, check, using P , whether any of these registers contains β . If so, reset these register contents to 0 and proceed, otherwise proceed without any modifications. This simulates Q on an unresetting α -machine.

To recognize limit steps in the computation of Q , reserve two extra registers, R_1 and R_2 ; initially, let R_1 contain 1 and R_2 contain 0. Whenever a step of Q is carried out, swap their contents. Whenever their contents are equal, set R_1 again to 1 and R_2 to 0. In this way, the contents of R_1 and R_2 will be equal iff the Q -computation has just reached a limit stage. \square

5.1. The unresetting case.

Lemma 34. Let $\alpha < \beta$ be ordinals. Then $wCOMP_\alpha \subseteq wCOMP_\beta \subseteq wCOMP_{\omega_1}$ and $wRECOG_\alpha \subseteq wRECOG_\beta \subseteq wRECOG_{\omega_1}$.

Proof. If $\alpha < \beta$, then terminating unresetting α -computations work exactly the same on unresetting β -machines. \square

We have seen above that $wCOMP_\omega = wRECOG_\omega$. We shall see now that that this happen again for ω_1 and in fact for all but countably many countable ordinals α .

Lemma 35. $wCOMP_{\omega_1} = wRECOG_{\omega_1}$.

Proof. This follows from Theorem 29, as $wCOMP_{\omega_1}$ and $wRECOG_{\omega_1}$ are just the set of *ORM*-computable and *ORM*-recognizable reals (without ordinal parameters), respectively. \square

Theorem 36. There is $\beta < \omega_1$ such that there are no lost melodies for unresetting γ -machines whenever $\gamma \geq \beta$.

Proof. Let β be large enough such that $wCOMP_{\beta} = wCOMP_{\omega_1}$ and $wRECOG_{\beta} = wRECOG_{\omega_1}$. (This is possible by monotonicity and the fact that there are only countably many programs.) Then, for all $\gamma \geq \beta$, we have $wCOMP_{\gamma} = wCOMP_{\omega_1} = wRECOG_{\omega_1} = wRECOG_{\gamma}$ by Lemma 35. \square

Our next goal is to show that there are ordinals α for which $wCOMP_{\alpha} \subsetneq wRECOG_{\alpha}$, i.e. for which the lost melody phenomenon does occur:

Lemma 37. There exists an unresetting $\omega + 1$ -program P such that $P(x) \downarrow = 1$ iff $x = \omega$ and $P(x) \downarrow = 0$, otherwise.

Proof. Let R_1 be the register containing x . Use a register R_2 to successively count upwards from 0. Use a flag to check whether the machine is in a limit state. Eventually, the content of R_1 is reached. If this happens in a limit step, then R_1 contains ω , otherwise, it does not. \square

Lemma 38. $wCOMP_{\omega+1} = COMP_{\omega}$ and $wRECOG_{\omega+1} = RECOG_{\omega}$, i.e. unresetting $\omega + 1$ -machines are equivalent in computational and recognizability strength to *ITRMs*.

Proof. (Sketch) One direction follows from Lemma 37 and Lemma 33.

For the other direction, we show that a resetting ω -machine (i.e. an *ITRM*) can simulate an unresetting $(\omega + 1)$ -machine. To see this, proceed as follows: Let P be a program for an unresetting $\omega + 1$ -machine. Assume that P uses k registers R'_1, \dots, R'_k . We set up an *ITRM*-program in the following way: Reserve R_1, \dots, R_k for the simulation of P . In the simulation, let 0 represent ω and let $i + 1$ represent i for all $i \in \omega \setminus \{0\}$. Whenever P requires that the content of R'_i is set to the value 0, set R_i to 1. When P requires that the content of R'_i is increased by 1 and this content is currently 0, stop. Otherwise, run P on R_1, \dots, R_k in the usual way. \square

Theorem 39. $wCOMP_{\omega+1} \neq wRECOG_{\omega+1}$, i.e. there are lost melodies for unresetting $\omega + 1$ -machines.

Proof. This follows immediately from Lemma 38, since, by the lost melody theorem for *ITRMs*, we have $COMP_{\omega} \neq RECOG_{\omega}$. \square

Remark 40. Arguments similar to the proof of Lemma 38 show that a resetting ω -machine can in fact simulate an unresetting $(\omega + i)$ -machine for all $i \in \omega$ (and more). On the other hand, it can be shown that this is no longer the case for unresetting α -machines when $\alpha > \omega_{\omega}^{CK}$ is exponentially closed: Coding $x \in L_{\alpha}$, $x = \{y \in L_{\beta} \mid L_{\beta} \models \phi_n(y, \vec{\gamma})\}$ (where $\vec{\beta}$

is a finite sequence of ordinals and $\beta < \alpha$) by $(\alpha, n, \vec{\gamma})$ and using techniques similar to those developed in [KoSy], we can evaluate arbitrary statements about the coded elements inside L_α with an unresetting α -machine. This allows us to search through $\mathcal{P}(\omega) \cap L_\alpha$ with such a machine. As in the proof of Theorem 29, one can now conclude that all reals in L_α recognizable by an unresetting α -machine are already computable by such a machine. We also saw that $RECOG_\omega \subseteq wRECOG_\alpha$ for $\alpha > \omega$. Now, the minimal real code $c := cc(L_{\omega_\omega^{CK}})$ of $L_{\omega_\omega^{CK}}$ is an element of $L_{\omega_\omega^{CK}+2}$, and hence of L_α . c is easily seen to be *ITRM*-recognizable, but as $c \notin L_{\omega_\omega^{CK}}$, it is not *ITRM*-computable. But $c \in RECOG_\omega \cap L_\alpha \subseteq wRECOG_\alpha \cap L_\alpha \subseteq wCOMP_\alpha$. So $c \in wCOMP_\alpha - COMP_\omega$.

Question: We saw that $wCOMP_{\omega+1} \subsetneq wRECOG_{\omega+1}$ and there is a countable β such that $wCOMP_\gamma = wRECOG_\gamma$ when $\gamma > \beta$. We do not know if there are gaps in the ordinals for which lost melodies exist, i.e. if there are $\omega + 1 < \gamma < \delta$ such that $wCOMP_\gamma = wRECOG_\gamma$, but $wCOMP_\delta \subsetneq wRECOG_\delta$.

5.2. The resetting case. Note first that the computational strength for various values of α much higher than in the unresetting case:

Theorem 41. Let P_i be some natural enumeration of the *ORM*-programs. There is $\alpha < \omega_1$ such that some resetting α -machine can solve the halting problem for parameter-free *ORM*s (i.e. unresetting ω_1 -machines), i.e. there is an unresetting α -program Q such that $Q(i) \downarrow = 1$ iff $P_i(0)$ stops and $Q(i) \downarrow = 0$ iff $P_i(0)$ diverges.

Proof. Let α_1 be the supremum of the register contents occurring in any halting parameter-free *ORM*-computation, let α_2 be the supremum of the parameter-free *ORM*-halting times and let $\alpha := \max\{\alpha_1, \alpha_2\}$ (of course, as all registers are initially 0 and a register content can be increased at most by 1 in one step, we will have $\alpha_1 \leq \alpha_2$; it is not hard to see that in fact $\alpha_1 = \alpha_2$).

Now consider *ORM*-programs with a fixed number n of registers. Then a resetting α -machine can solve the halting problem for such programs by simply simulating the given program P in the registers R_1, \dots, R_n , while using a further register R_{n+1} as a clock by increasing its content by 1 whenever a step in the simulation is carried out. If any of the registers R_1, \dots, R_n, R_{n+1} overflows, then P does not halt and we output 0; otherwise, the simulation reaches the halting state and we output 1.

A register overflow can be detected as follows: If a register R has overflowed, then the machine must be in a limit state (which can be detected by flags in the usual way) and R must contain 0. In this situation, either there has been an overflow or the prior content of R has been 0 cofinally often in the current running time. This can be

distinguished by an extra register R' whose content is set to 0 whenever R contains 0 and to 1, otherwise. Hence, if R' contains 0 in a limit state, then the content of R must have been 0 cofinally often.

Now, by [KoSy], there is a universal *ORM*, so we have an effective method how to find, for every *ORM*-program P , an *ORM*-program with the same halting behaviour, but using only 12 registers. This, in combination with the halting problem solver for programs with any fixed number of registers, solves the halting problem for *ORMs*. \square

The same holds when one considers the recognizability strength. To show this, we need some preliminaries.

Lemma 42. There is an *ITRM*-program R such that, for each real x coding an ordinal $\alpha < \omega_1$ according to $f : \alpha \rightarrow \omega$ injective, R^x changes the content of the register R_1 exactly $\alpha + 1$ many times.

Proof. By Lemma 2 of [KoMi], the program P defined there to test the oracle for well-foundedness of the coded relation runs for at least β many steps when the oracle codes a well-ordering of length β . Roughly, P uses a stack to represent a finite descending sequence and attempts to continue it. We reserve a separate register R_1 and flip its content between 0 and 1 whenever a new element is put on the stack in the computation of P^x . The argument for Lemma 2 of [KoMi] shows that the content of R_1 will be changed at least α many times. If this happens exactly α many times, we simply set up our program to flip the content of R_1 once more after P has stopped. If it happens more than α many times, then some finite sequence \vec{s} of natural numbers is the $\alpha + 1$ th sequence that is put on the stack and we set up our program to stop once \vec{s} has appeared on the stack. \square

Corollary 43. Let $\alpha < \omega_1$. There is a resetting α -program I which, given a real x coding an ordinal γ , halts with output 1 iff $\gamma = \alpha$ and otherwise halts with output 0.

Proof. As α -register machines can simulate *ITRMs*, we can use Lemma 42 to obtain a program R that (run in the oracle x) changes the content of register R_1 exactly $\alpha + 1$ many times. We use a separate register R_2 that starts with content 0 and whose content is incremented by 1 whenever the content of R_1 is changed. Now, if R_2 overflows and the content of R_1 is changed afterwards without R halting, then $\alpha < \gamma$. If, on the other hand, R stops without R_2 having overflowed, we have $\gamma < \alpha$. If neither happens, i.e. if R_2 overflows and the next change of the content of R_1 is followed by R halting, then $\alpha = \gamma$. These scenarios are easy to detect. \square

Theorem 44. There exists $\alpha < \omega_1$ and $x \subseteq \omega$ such that $x \in \text{RECOG}_\alpha$, but $x \notin \text{wRECOG}_{\omega_1}$.

Proof. Let τ be the supremum of stages containing new *ORM*-recognizables. Let $\alpha + 1 > \tau$ be an index such that $L_\alpha \models ZF^-$ and let $r := cc(L_\alpha)$ be the $<_L$ -minimal real coding L_α . It is well known that this implies $cc(L_\alpha) \in L_{\alpha+2}$ (see e.g. [BoPu]). Then r is recognizable by a resetting α -machine. To see this, first note that the property of being the minimal code of an index L -stage can be checked by an *ITRM* using the strategy described in the proof of the lost melody theorem for *ITRMs* in [ITRM]. We saw above that resetting α -machines can simulate *ITRMs* for all $\alpha \geq \omega$, hence this can be carried out by a resetting $\alpha + 1$ -machine. It only remains to test whether the coded stage L_ζ is indeed L_α . This can be done by using Corollary 43 to test whether the order type of $On \cap L_\zeta$ is α . \square

Theorem 45. Let $\alpha < \omega_1$, and let $\delta > \alpha$ be such that δ is a limit of indices, but not itself an index. Then any α -machine computation (with empty input and oracle) either halts in less than δ many steps or does not halt at all.

Proof. This is an adaption of the argument given in [KoMi] for *ITRMs*. As δ is a limit of indices, but not an index, it follows (see e.g. [Ch], [MaSr] or [LePu]) that $L_\delta \models ZF^- + \forall x \exists f(f : \omega \rightarrow_{surj} x)$ and hence that (see [Je]) $\rho_\omega^\delta = \delta$, where ρ_ω^α denotes the ultimate projectum of L_α . We claim that there is no $f : \xi \rightarrow \delta$ with unbounded range and $\xi < \delta$ definable over L_δ . To see this, assume that there is such an f . By assumption, there is, for every $\beta < \delta$ an index between β and δ and hence L_δ contains a $<_L$ -minimal bijection g_β between ω and β . Define a map $\tilde{f} : \xi \times \omega \rightarrow_{surj} \delta$ via $\tilde{f}(\iota, j) = g_{f(\iota)}(j)$. Let h be a bijection between $\xi \times \omega$ and $\xi\omega$ and define $\tilde{f} : \xi\omega \rightarrow_{surj} \delta$ by $\tilde{f} := \tilde{f} \circ h^{-1}$. As $\xi < \delta$ and $L_\delta \models ZF^-$, we also have $\xi\omega < \delta$, and \tilde{f} is certainly definable over L_δ . Hence a surjection from some $\zeta < \delta$ onto δ (and hence onto L_δ) is definable over L_δ , so that $\rho_\omega^\delta < \delta = \rho_\omega^\delta$, a contradiction.

Now, there is a natural injection from the states of an α -machine into α^ω , as the state can be given by a finite tuple of ordinals $< \alpha$ representing the register contents and a single natural number representing the active program line. Such a map j is definable over L_{α^ω} and hence certainly an element of L_δ .

Now let P be an α -program, and let C be the computation of P , restricted to the first δ many steps. For a machine state s , let γ_s denote $\sup\{\beta < \delta \mid C(\beta) = s\}$.

Assume first that $\{\beta < \delta \mid \gamma_{C(\beta)} < \delta\}$ is cofinal in δ , i.e. there are cofinally many states that appear only on boundedly many times. Then we can define, over L_δ , a partial map $f : \alpha\omega \rightarrow \delta$ by letting $f(\xi) = \gamma_{j^{-1}(\xi)}$ if $j^{-1}(\xi)$ is defined and $\gamma_{f^{-1}(\xi)} < \delta$ and otherwise $f(\xi) = 0$. By assumption, f has unbounded range in δ , which contradicts our observation above.

Hence, we may assume that there is some $\gamma < \delta$ such that every machine state assumed after time γ appears at cofinally in δ many times. Suppose that P uses $n \in \omega$ many registers. The possible machine states are hence elements of $\times_{i=1}^n \alpha \times \omega$. Let us partially order the set S of machine states occurring in the computation after time γ by letting $(\beta_1, \dots, \beta_n, k) \leq_s (\gamma_1, \dots, \gamma_n, l)$ iff $k \leq l$ and $\beta_i \leq \gamma_i$ for all $i \in \{1, \dots, n\}$. It is easy to see that \leq_s is well-founded.

For each two states $Z_1, Z_2 \in S$, there is $Z_3 \in S$ such that $Z_1 \leq_s Z_3$ and $Z_2 \leq_s Z_3$: To see this, observe that we can define over L_δ a strictly increasing map $\sigma : \omega \rightarrow \delta$ such that $C(\sigma(2i)) = Z_1$ and $C(\sigma(2i+1)) = Z_2$ for all $i \in \omega$. By our observation above, $\text{rng}(\sigma)$ must be bounded in δ , so let $\bar{\delta} := \sup \text{rng}(\sigma)$. Then $C(\bar{\delta})$ is as desired.

Now, by well-foundedness of \leq_s , S must contain a minimal element Z . It is easy to see that Z is in fact unique: For if Z_1 and Z_2 were two distinct minimal elements of S , then by our last observation, we would have $Z_3 \in S$ with $Z_3 \leq Z_1$ and $Z_3 \leq Z_2$. As $Z_1 \neq Z_2$, one of the inequalities would have to be strict, contradicting the minimality of Z_1 and Z_2 .

Hence Z is assumed cofinally in δ many times, while all other states occurring after time γ are $\geq_s Z$. Consequently, the machine state at time δ is again Z and it is easy to see that the computation cycles. Hence, a resetting α -machine computation either halts before time δ or does not halt at all. □

Corollary 46. $COMP_\alpha \subseteq L_\delta$, where δ is the minimal limit of indices above α which is not itself an index.

Proof. Since δ is not an index, every subset of ω definable over L_δ is an element of L_δ . Now let $x \in COMP_\alpha$, and let P be a resetting α -program that computes x , i.e. $P(i) \downarrow = 1$ if $i \in x$ and $P(i) \downarrow = 0$ if $i \notin x$ for all $i \in \omega$. By Theorem 45 and as $P(i) \downarrow$ for all $i \in \omega$, the halting time of $P(i)$ must be smaller than δ for all $i \in \omega$. Hence $i \in x$ is expressed over L_δ by an \in -formula stating the existence of a halting P -computation with input i and output 1. Consequently, we must have $x \in L_\delta$. □

This allows us to show that there are lost melodies for resetting α -machines for all infinite $\alpha < \omega_1$:

Theorem 47. Let $\alpha < \omega_1$ be infinite. Then there $COMP_\alpha \neq RECOG_\alpha$, i.e. there is a lost melody for resetting α -machines.

Proof. Given $\alpha < \omega_1$, let r_α be the $<_L$ -minimal real coding an L -level L_γ such that γ is a limit of indices but not itself an index, $\gamma + 1$ is an index and L_γ contains a real coding α . Then we must also have $r_\alpha \notin COMP_\alpha$ by Corollary 46. We show that $r_\alpha \in RECOG_\alpha$ by an argument similar to the proof of the lost melody theorem for *ITRMs*.

Let x be given in the oracle. First, we can - even with an *ITRM* - check whether x codes an L -level L_ζ with cofinally many indices. If not, $x \neq r_\alpha$. If so, the methods developed in the proof of the lost melody theorem for *ITRMs* allow us to compute from x the truth predicate for $L_{\zeta+2}$, which allows us to check whether ζ and $\zeta + 1$ are indices. If ζ is an index or $\zeta + 1$ is not, then $x \neq r_\alpha$. Otherwise, we need to check whether $\zeta > \alpha$ (this suffices to guarantee the existence of a real coding α , since at this point we already know that ζ is a limit of indices). This can be done as follows: Inside r_α , α must be coded by some natural number i that can be given to our program in advance. So we test whether i codes an ordinal θ in x . If not, then $x \neq r_\alpha$. Now, we can easily compute from i and x a real y coding the order type θ (just delete every $p(k, j) \in x$ with $\{p(k, i), p(j, i)\} \not\subseteq x$) and then use Corollary 43 to check whether y codes α . If not, then $x \neq r_\alpha$. Otherwise, we know that i codes α inside r_α .

Next, we check whether there is any $\alpha < \zeta' < \zeta$ with the same properties. If yes, then $x \neq r_\alpha$. Otherwise, we know that x codes L_γ and it remains to check the $<_L$ -minimality of x . As $L_{\zeta+1}$ is an index, we know that the minimal real coding L_ζ must be an element of $L_{\zeta+2}$. As we just mentioned, we can, given x , evaluate the truth predicate for $L_{\zeta+2}$. Hence, we can search through (the code of) $L_{\zeta+2}$ until we find the $<_L$ -minimal real coding L_ζ and compare it with x . If these reals disagree, then $x \neq r_\alpha$, otherwise $x = r_\alpha$. So r_α is recognizable.

This proves that r_α is a lost melody for resetting α -machines.

It remains to see that such an L -level L_γ exists. To see this, let $\gamma > \alpha$ be a minimal limit of indices, and let $\alpha < \delta < \gamma$ be an index. Let x be a real such that $x \in L_{\delta+1} - L_\delta$. Then the elementary hull H of $\{x\}$ in L_γ is (isomorphic to) an L -level L_β with cofinally many indices which contains x , where $\beta \leq \gamma$. It follows that $\beta = \gamma$ and that in fact $H = L_\gamma$. This hull is definable over $L_{\gamma+1}$, so that we get a bijection between ω and L_γ in $L_{\gamma+2}$ by the standard finestructural arguments. Hence $\gamma + 1$ is indeed an index, so γ is as desired. \square

Remark 48. Note that, as parameter-free computations have countable length, $wCOMP_{\omega_1}$ corresponds to parameter-free *ORM*-computability. Moreover, by Shoenfield absoluteness, we have $wRECOG_{\omega_1} \subseteq wRECOG_{\omega_1} \subseteq \mathcal{P}^L(\omega)$. Consequently, if P is an *ORM*-program recognizing $x \subseteq \omega$, then P , now interpreted as a program for an unresetting ω_1 -ITRM, recognizes x as well: The computations will only take countable many steps and hence no limit of register contents can exceed ω_1 , prompting an overflow. Hence $wRECOG_{\omega_1}$ coincides with the set of *ORM*-recognizable reals. The same holds for every $\alpha \geq \omega_1$. As *ORM*-computability and *ORM*-computability coincide, there are no lost melodies for unresetting α -ITRMs with $\alpha \geq \omega_1$.

6. CONCLUSION AND FURTHER WORK

We have seen that lost melodies exist for a resetting α -machines iff $\alpha < \omega_1$ is infinite and that for unresetting α -machines, lost melodies do not exist for $\alpha = \omega$, do exist for $\alpha = \omega + 1$ and cease to exist from some countable ordinal on. In the special case of resetting ω -machines or *ITRMs*, the recognizable allow for a detailed analysis among the constructible reals and show several surprising regularities. In the parameter-*OTM*-case, we reach the limits of *ZFC*. In general, the relation between the computability and recognizability strength of transfinite models of computation seems to be far from trivial.

In this paper, we have restricted our attention to reals, as these can be dealt with by all models in question and can hence be used as a basis for comparison. One could equally well consider subsets of other ordinals, which might be more appropriate for some models.

Once we do this, interesting questions arise, even for classical Turing machines: Consider, for example, Turing programs using at most n states and symbols for some $n \in \mathbb{N}$. Let us say that a natural number k is n -computable iff there is such a Turing program that outputs k when run on the empty input, and let us say that k is n -recognizable iff there is such a Turing program that stops with output 1 on the input k and with output 0 on all other integers. Are there infinitely many $n \in \mathbb{N}$ for which there exists $l \in \mathbb{N}$ which is n -recognizable, but not n -computable? This provides a kind of a miniaturization of the question for the existence of lost melodies.

Another topic one might pursue is to consider the various generalizations of Turing machines (*ITTM*s, α -Turing machines, α - β -Turing machines).

7. ACKNOWLEDGMENTS

We are indebted to Philipp Schlicht for sketching a proof of Lemma 5, a crucial hint for the proof of Theorem 8 and suggesting several very helpful references, in particular concerning Theorem 27. We also thank the anonymous referee for several corrections and suggestions that helped to considerably improve the paper.

REFERENCES

- [Ba] J. Barwise. Admissible Sets and Structures. Springer (1975)
- [BoPu] G. Boolos, H. Putnam. Degrees of unsolvability of constructible sets of integers. *Journal of Symbolic Logic*, 37 (1972), 81-89
- [Ca] M. Carl. The distribution of *ITRM*-recognizable reals. To appear in: *Annals of Pure and Applied Logic*, special issue from the Turing Centenary Conference CiE 2012: How the World Computes
- [Ca2] M. Carl. Optimal results on *ITRM*-recognizability. Preprint. arXiv:1306.5128v1 [math.LO]

- [Ch] C.T. Chong. A recursion-theoretic characterization of constructible reals. Bulletin of the London Mathematical Society 9, 241-244 (1977)
- [Ko] P. Koepke. Computing a model of set theory. In New Computational Paradigms. S. Barry Cooper et al, eds., Lecture Notes in Computer Science 3988 (2006), 223-232
- [Co] S. Coskey. Infinite-Time Turing Machines and Borel Reducibility. Mathematical Theory and Computational Practice (2009)
- [Cu] N. Cutland. Computability. An introduction to recursive function theory. Cambridge University Press (1980)
- [FrWe] S.D. Friedman, P. D. Welch. J. Symbolic Logic Volume 76, Issue 2 (2011), 620-636.
- [GHJ] V. Gitman, J.D. Hamkins, Th. A. Johnstone. What is the theory ZFC without power set? arXiv:1110.2430 [math.LO]
- [HaLe] J. D. Hamkins, A. Lewis. Infinite Time Turing Machines. Journal of Symbolic Logic 65(2), 567-604 (2000)
- [HMSW] J. D. Hamkins, R. Miller, D. Seabold, and S. Warner. Infinite time computable model theory. , S. B. Cooper, B. Löwe, and A. Sorbi, Ed., New York: Springer, 2008, pp. 521-557.
- [ICTT] M. Carl. Towards a Church-Turing-Thesis for Infinitary Computations. Electronic Proceedings of CiE 2013.
- [ITRM] M. Carl, T. Fischbach, P. Koepke, R. Miller, M. Nasfi, G. Weckbecker. The basic theory of infinite time register machines. Archive for Mathematical Logic 49 (2010) 2, 249-273
- [Je] R.B. Jensen. The fine structure of the constructible hierarchy. Annals of Mathematical Logic 4 (1972)
- [JeKa] R. Jensen, C. Karp. Primitive Recursive Set Functions. In: Proceedings of Symposia in Pure Mathematics, Volume XIII, Part 1 (1971)
- [Ka] A. Kanamori. The higher infinite. Springer 2005.
- [Ko] P. Koepke. Computing a model of set theory. In New Computational Paradigms. S. Barry Cooper et al, eds., Lecture Notes in Computer Science 3988 (2006), 223-232
- [KoSe1] P. Koepke, B. Seyffert. Ordinal machines and admissible recursion theory. Annals of Pure and Applied Logic, 160 (2009), 310-318.
- [KoSe2] P. Koepke, Benjamin Seyffert. Towards a theory of infinite time Blum-Shub-Smale machines. 7 pages. Proceedings of CiE 2012
- [KoMi] P. Koepke, Russell Miller. An enhanced theory of infinite time register machines. In Logic and Theory of Algorithms. A. Beckmann et al, eds., Lecture Notes in Computer Science 5028 (2008), 306-315
- [KoSy] P. Koepke, R. Syders. Register Computations on Ordinals. Archive for Mathematical Logic 47 (2008), 529-548
- [KoWe] P. Koepke, P. Welch. A generalised dynamical system, infinite time register machines, and $Pi_1^1 - CA_0$. In CiE 2011. B. Löwe et al, eds., Lecture Notes in Computer Science 6735 (2011), 152-159.
- [LePu] S. Leeds, H. Putnam. An intrinsic characterization of the hierarchy of constructible sets of integers. Logic Colloquium '69 (North-Holland, Amsterdam, 1971)
- [MaSr] W. Marek, M. Srebrny. Gaps in the constructible universe. Ann. Math. Logic, 6 (1974), 359-394
- [Ma] A.R.D. Mathias. Provident sets and rudimentary set forcing. Preprint, available at <https://www.dpmms.cam.ac.uk/~ardm/>

- [ORM] P. Koepke. R. Syders. Computing the recursive truth predicate on ordinal register machines. In Logical Approaches to Computational Barriers, Arnold Beckmann et al., eds., Computer Science Report Series 7 (2006), Swansea, 160-169
- [OTM] P. Koepke. Turing computations on ordinals. Bulletin of Symbolic Logic 11 (2005), 377-397
- [Sa] G. Sacks. Higher recursion theory. Springer 1990.
- [Sa2] G. Sacks. Countable admissible ordinals and hyperdegrees. Advances in Mathematics 19, 213-262 (1976)
- [Se] B. Seyfferth. Three models of ordinal computability, PhD thesis, University of Bonn, 2012
- [SeSc] P. Schlicht, B. Seyfferth. Tree representations via ordinal machines. Computability 1, 1 (2012), 45-57
- [wITRM] P. Koepke. Infinite Time Register Machines. Logical Approaches to Computational Barriers, Arnold Beckmann et al., eds., Lecture Notes in Computer Science 3988 (2006), 257-266